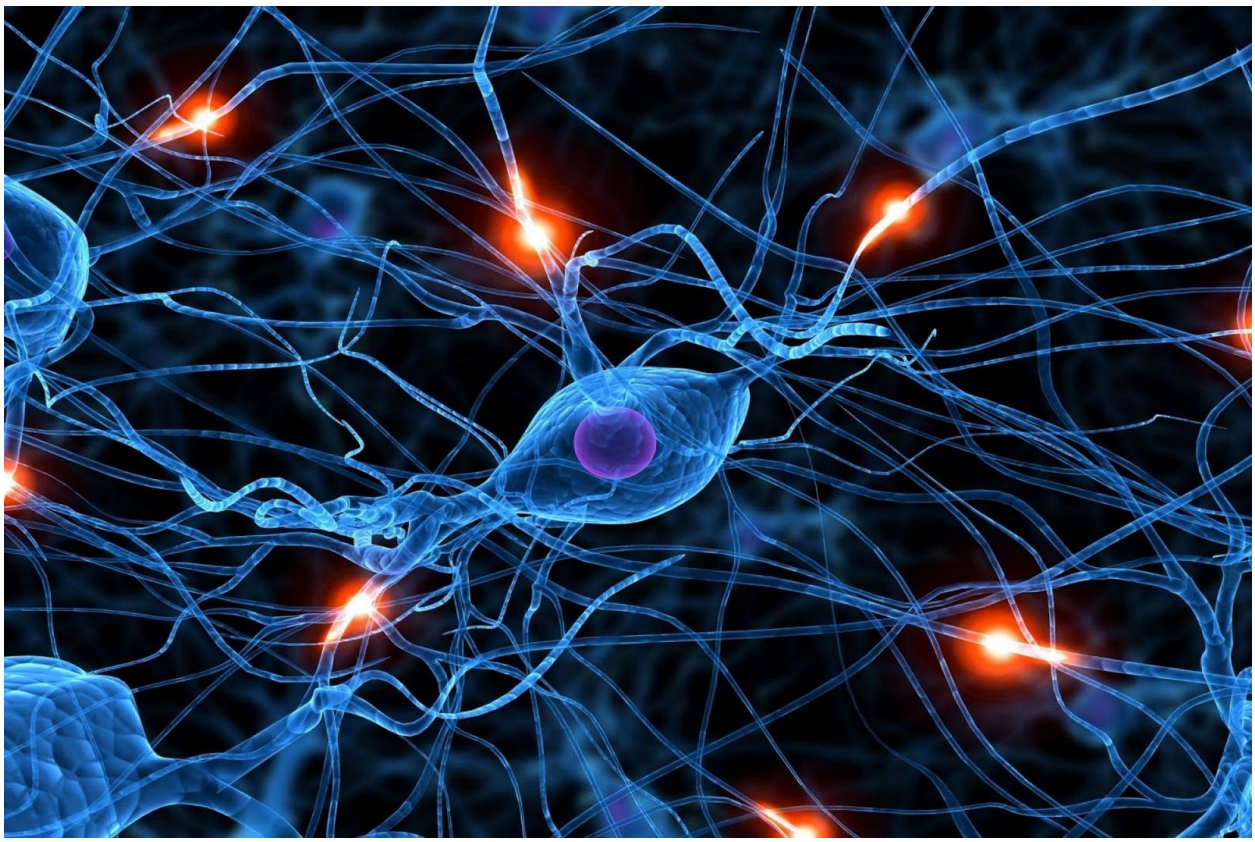
Sistemas de Inteligencia Artificial

Trabajo Práctico:
Redes Neuronales



Integrantes:

Juan Franco Caracciolo - 56382

Facundo González Fernández - 55746

Julian Nicastro - 55291

Sebastian Ezequiel Guido - 54432

Introducción

El objetivo de este trabajo práctico es encontrar una arquitectura óptima de una red neuronal para resolver un problema de generación de terrenos. Para esto se deberá entrenar la red con distintas arquitecturas y con distintas metodologías, funciones de activación, parámetros de entrenamiento y diversas optimizaciones.

Conceptos claves

Backpropagation

Se basa en emplear un ciclo propagación – adaptación de dos fases. Una vez que se ha aplicado un patrón a la entrada de la red, se propaga desde la primera capa a través de las capas siguientes, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

Red Multicapa

Se utilizó una red multicapa ya que, con esta arquitectura, se pueden resolver problemas que no son linealmente separables, lo cual es la principal limitación del perceptrón simple.

Funciones de Activación

Función que define el output de una neurona dado un input y su peso. Se utilizaron dos tipos de funciones de activación, exponencial y tangencial.

Optimizaciones

Se realizaron dos optimizaciones importantes. La primera es Momentum y la segunda es de Parámetros para el eta Adaptativos. Estas optimizaciones permiten una convergencia más rápida como también la posibilidad de saltar y escapar de mínimos locales. Si bien solamente expondremos los resultados de estas dos optimizaciones, también se realizaron algunas otras en cuestiones de operaciones aritméticas propias de octave.

Momentum

Consiste en agregar un término a la función de corrección de pesos, durante el aprendizaje. El objetivo es evitar cambios bruscos en la orientación del aprendizaje para que, así, converja más rápido. Se pudo apreciar un cambio notable luego de la implementación. A continuación se pueden apreciar la diferencia de aplicar esta optimización.

Parámetros Adaptativos

Se basa en la modificación de una variable ETA (originalmente el learning rate), que acelera o desacelera el aprendizaje. Se propone que, en el caso que la red esté aprendiendo y disminuyendo su tasa de error, se acelere el aprendizaje aumentando el ETA. Y, en el caso que se haya acelerado demasiado y se vea que la tasa de error empiece a aumentar, desacelerar el aprendizaje disminuyendo ETA.

Modelos de Aprendizaje

A la hora de entrenar la red, se estudiaron dos enfoques distintos, los cuales varían únicamente en la selección del momento en el cual se realiza la corrección de los pesos en base a los errores cometidos por la red.

Incremental

En primer instancia, se analizó el método de entrenamiento incremental. Este método consiste en pasar cada patrón de entrenamiento por la red, calcular el error correspondiente e inmediatamente después corregir los pesos de toda la red en base a ello. Luego se repite el proceso con cada uno de los patrones restantes. De esta manera durante cada época se entrena varias veces la red, pero esto puede traer problemas a la hora de ver como el error progresa, por lo que en lugar de analizar cómo el mismo progreso a lo largo de cada iteración, se analizó la variación del error al finalizar cada época.

Batch

En cambio, la metodología de entrenamiento batch consiste en primero calcular el error para cada uno de los patrones de una época y realizar una única corrección durante la misma en base al error de todos ellos. Sin embargo, podría ocurrir que patrones que generan aportes opuestos a la nueva variacion en pesos y por lo tanto se anulen y la red no progrese.

Variacion de Parametros de Entrada

Hay varios parámetros iniciales que son cruciales en el proceso de aprendizaje. Entre ellas, se encuentra el Learning Rate y la arquitectura de la red. Distintos valores para los parámetros de entrada pueden lograr un aprendizaje exitoso como también uno fallido.

Learning Rate

Es un parámetro que representa la tasa de corrección de los pesos de la red neuronal. Cuanto mayor sea el Learning Rate, mayor será el grado de corrección, puede llegar a converger más rápido. Pero, a su vez, se puede estancar cuando se necesita modificar los pesos en menor proporción. Por otro lado, si el Learning rate es bajo, es más fácil para la red llegar a un mínimo error mas optimo, pero su convergencia es severamente más lenta

Estructura

Se denomina estructura de la red a la cantidad de capas y nodos internos (neuronas) en cada una de ellas. Las conexiones presentes entre neuronas se encuentran entre las neuronas de un nivel, o capa, inferior con el inmediatamente superior, en donde todas las neuronas de un nivel están conectadas con todas las del superior.

En el nivel inferior se cuenta con una neurona por cada valor de entrada que se le proporciona a la red, sumado a esto se contará con una neurona para actuar como threshold. En cada capa las neuronas superiores tendrán conexiones con todas las neuronas de la capa inferior salvo por la neurona que actúe como threshold, la cual solo se conectara hacia arriba con todas menos el threshold. De esta manera se conforma la red hasta llegar a la última capa con la neurona que provee la salida de la red.

A la hora de decidir la estructura de la red, es decir, la cantidad de capas ocultas y neuronas por capas con las que se iba a operar, se buscó extender el número de las mismas tanto vertical (es decir, incrementando el número de capas) como horizontalmente y observar de qué manera afecta esto a la red.

Se comenzó analizando el comportamiento de la red con pocas neuronas (2 o 3) y una única capa y que la red comenzaba aprendiendo pero rápidamente se estancaba en un nivel de errores elevado, incapaz de mejorar el aprendizaje sin mayores unidades neuronales.

Por lo tanto se prosiguió aumentando la cantidad de neuronas de esta capa y se halló que teniendo alrededor de 25 neuronas, se lograba reducir el error de entrenamiento cuadrático medio a valores menores a la diezmilésima, obteniendo una configuración válida para la red.

De todas formas se busco una configuración con dos capas, con una cantidad similar de neuronas, que pudiera reducir el error de la misma manera que la anteriormente mencionada. Se pudo hallar que una red de 10 y 15 neuronas en las capas ocultas alcanzaba los mismos errores que la de una capa de 25, pero en una menor cantidad de épocas de entrenamiento.

Aunque demorara una mayor cantidad de tiempo en entrenarse, se escogió la red de una sola capa, ya que la cantidad de operaciones para realizar cálculos con la misma luego es menor que su contraparte, y dado que ambas poseen errores similares, no es necesario decidir en base al tiempo que tomó entrenarlas.

Búsqueda de parámetros óptimos

Debido a la cantidad de parámetros que se tienen en cuenta que afectan de distintas manera a la red neuronal, probar distintos valores de parámetros es una tarea tediosa. Para esto buscamos maneras de optimizar el proceso.

Una alternativa relacionada a la materia es la de generar un algoritmo genético. Un algoritmo genético es aquel que saca sus principios de las observaciones de Darwin sobre la evolución y la supervivencia del más apto. Para esto se genera un pool de individuos, los cuales se ponen a prueba. Aquellos individuos con mejor rendimiento, sobreviven a la siguiente generación mientras que los menos afortunados quedaran atras. La siguiente generación entonces se agregaran individuos descendientes, los cuales se generarán

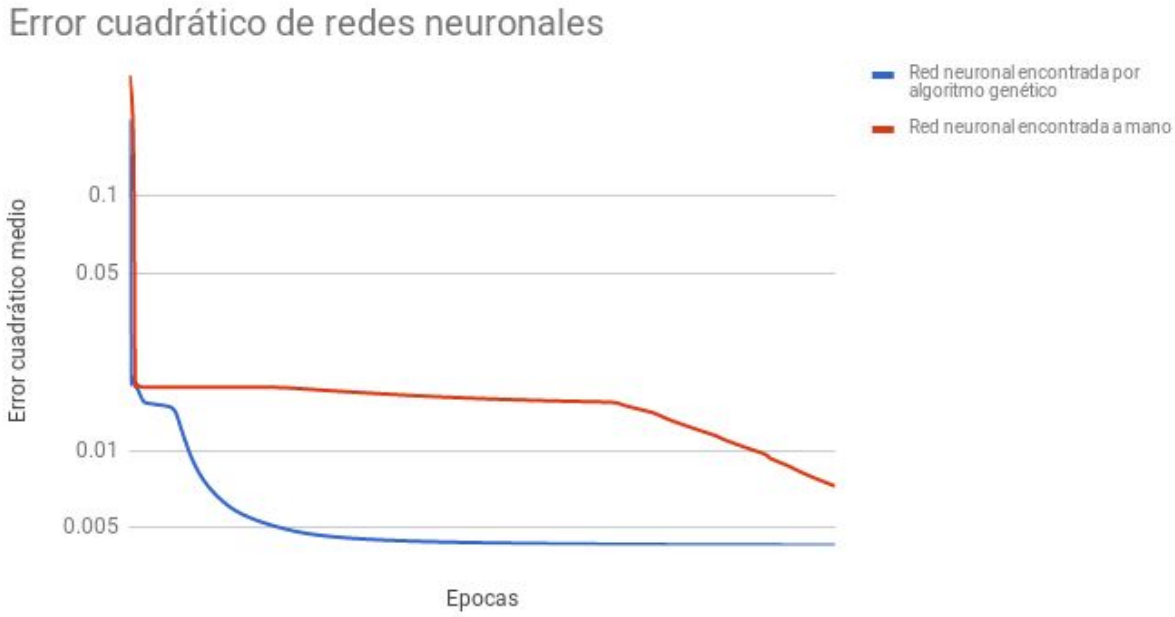
parecidos a sus padres pero con pequeñas diferencias. Aquellos con mejor rendimiento poseen mayor probabilidad de pasar sus genes a la próxima generación.

Nuestro conocimiento sobre este tipo de algoritmos es limitado pero parecía una alternativa muy apropiada para este problema. Se puede ver a una red neuronal como un conjunto de parámetros iniciales que la generan. Los parámetros que nos interesan variar son:

- **La estructura**
- **El learning Rate**
- **El alpha del momento**
- **El factor positivo del eta adaptativo**
- **EL factor negativo del eta adaptativo**

Entonces para definir el rendimiento de una red, se corre 3 veces con estos parámetros fijos, y se obtiene el promedio del error dado luego de una cantidad fija de épocas. Así, las configuraciones de redes que generaron los menores errores sobreviven a la siguiente generación y tendrían mayor posibilidad de pasar sus genes. La manera de generar descendencia es simplemente la de tomar dos configuraciones y generar para cada parámetro un nuevo valor que sea el promedio de ambos con un error aleatorio del 20%. Asumimos que estos 5 parámetros poseían una convergencia lineal hacia mínimos locales, lo que se vería en una descendencia del error promedio por generación. Pudimos entonces encontrar que esto de hecho pas


Error promedio por generación
0.016
0.015
0.014
0.013
0.012
Error promedio
5
10
15
20
Generación
Iteracion 1
Iteracion 2
Iteracion 3

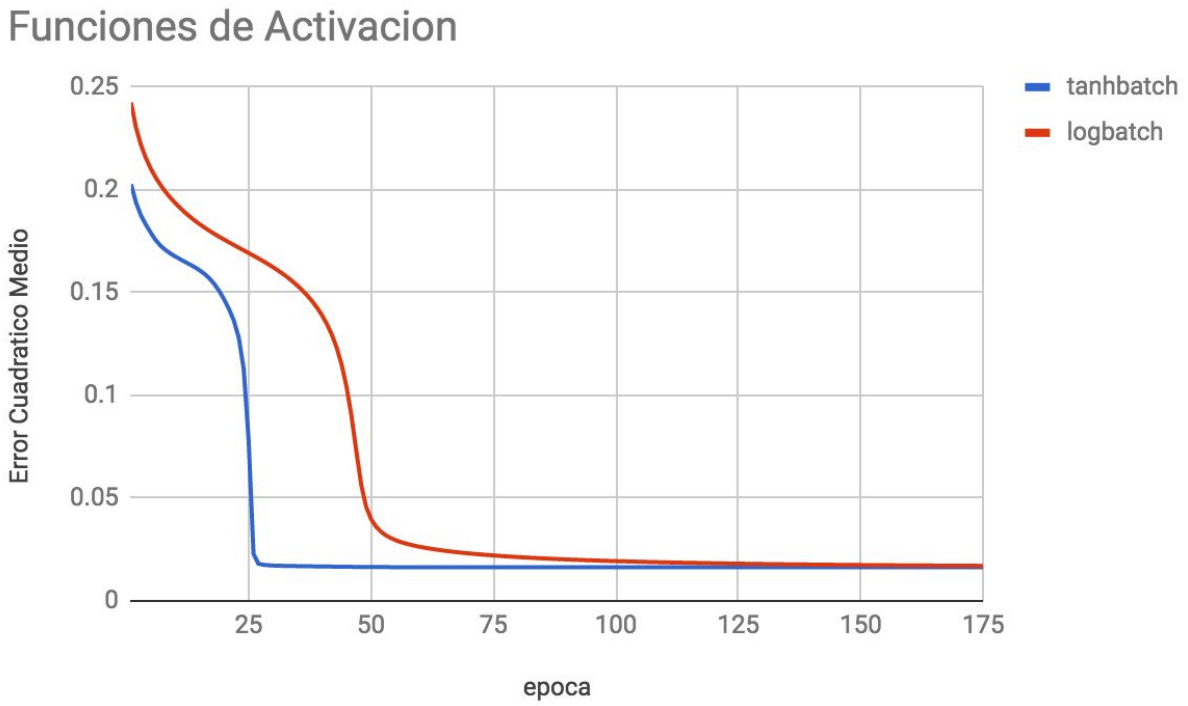

Error cuadrático de redes neuronales
0.1
0.05
0.01
0.005
Error cuadrático medio
Epocas
Red neuronal encontrada por algoritmo genético
Red neuronal encontrada a mano

Observaciones

Entrenamiento batch vs Incremental

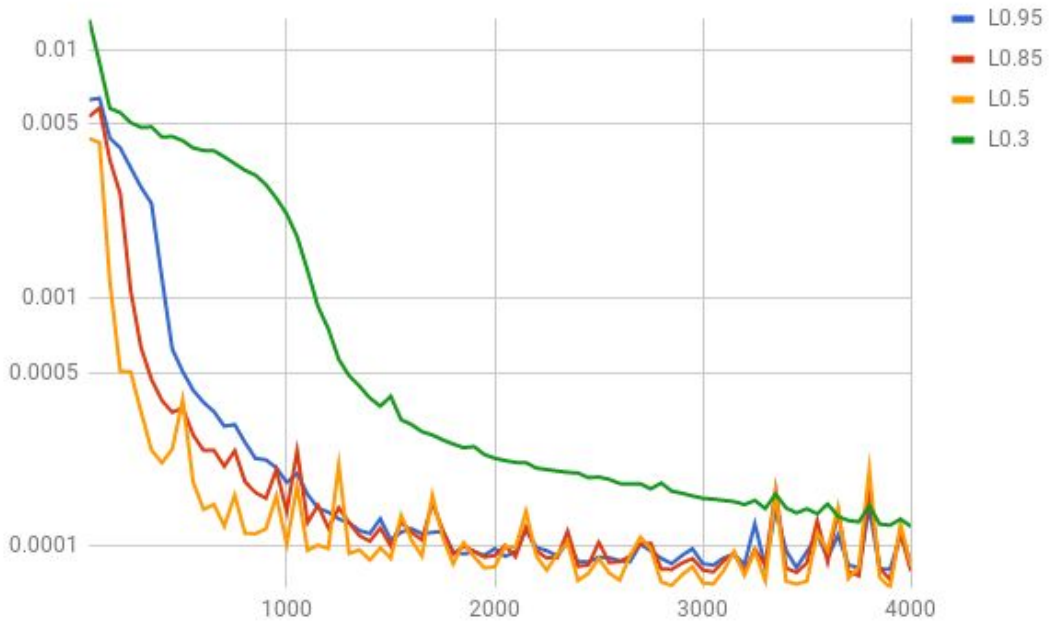
Pudimos realizar el cálculo tanto en metodología batch como en metodología incremental. Debido a su característica de cómputo, pudiendo hacer toda la gama de patrones con un simple cálculo de matrices, las épocas de batch eran considerablemente más rápidas que la incremental. Sin embargo, al sólo modificar la red una vez por época, las épocas generaban un menor cambio en la red, mostrando así una convergencia más lenta por época que el incremental.

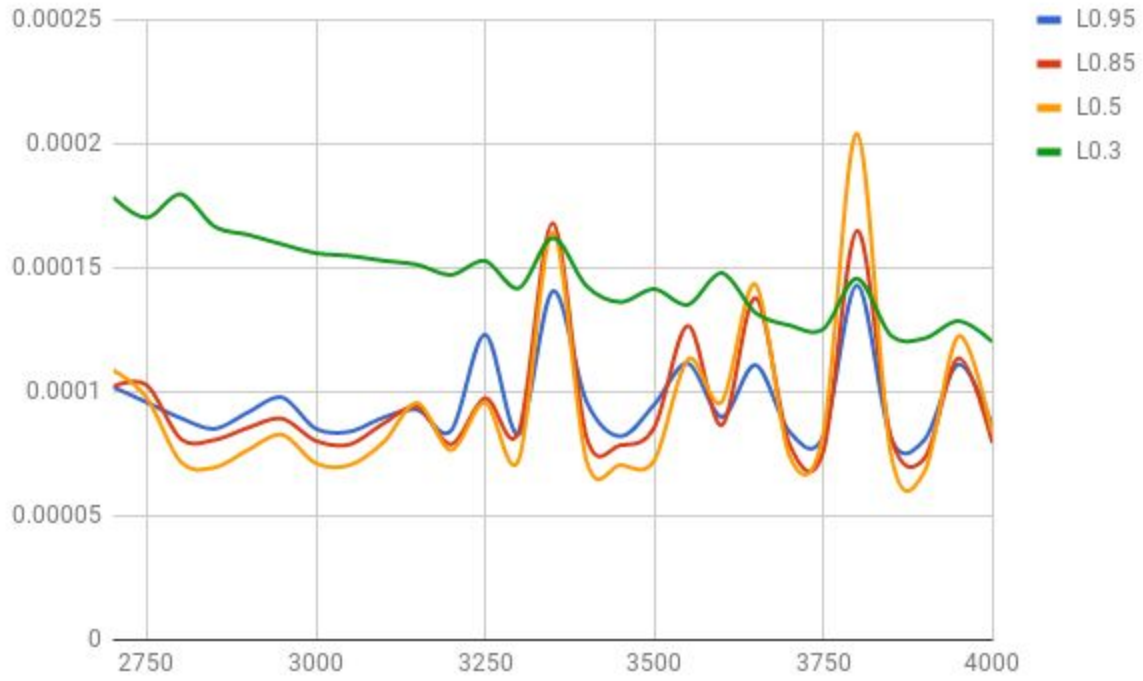
Funciones de Activación



La función tanH se ve que converge más rápido que la sigmoide por lo general. Para poder hacer que la red aprendiera con tanH como función de activación, el output que debía aprender era normalizado a valores de 0-1, y la función era modificada para que su imagen también fuera de 0-1.

Variación del Learning Rate

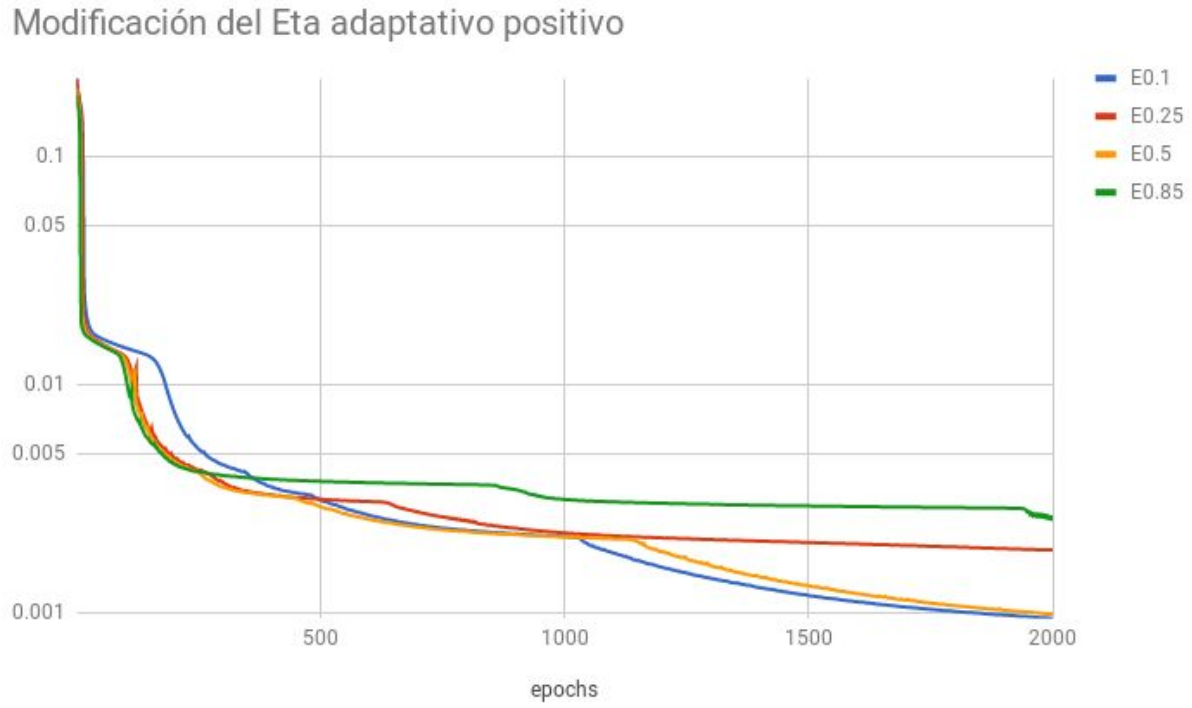


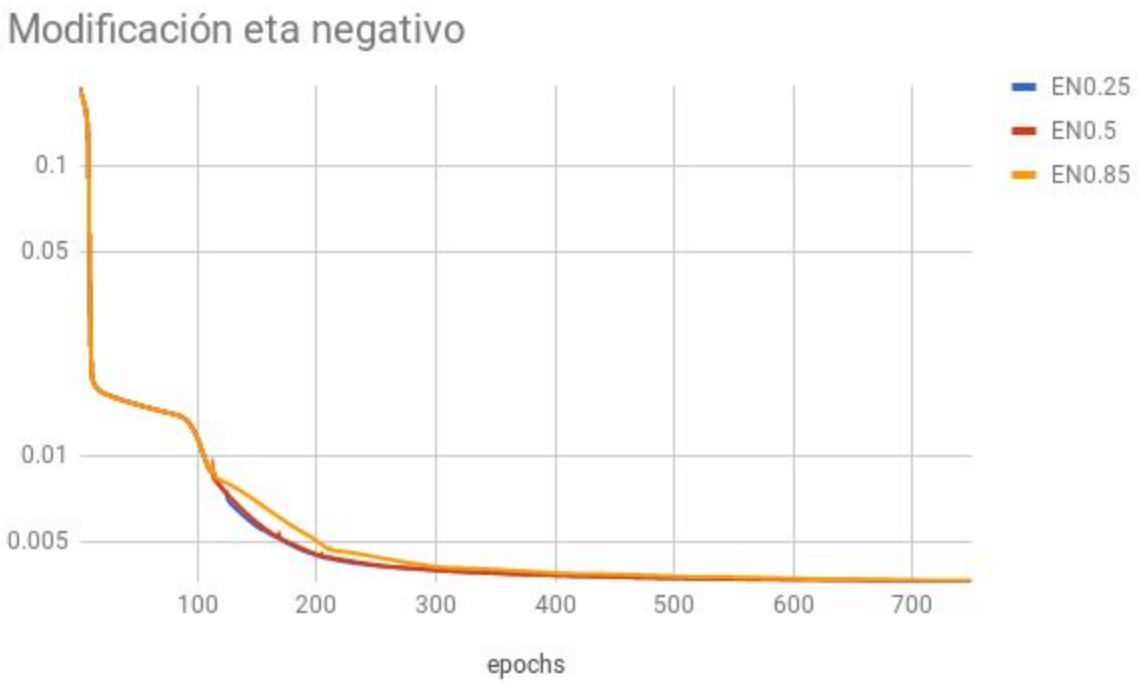


Se puede ver como con un learning rate menor la red demora mucho más en converger, pero tambien notamos que el error es más errático, ya que se disminuye el cambio que se

realiza en cada iteración de aprendizaje. Se puede notar también como aquellos con learning rate mayor tienden a estancarse erráticamente mientras el learning rate menor continúa bajando. ,

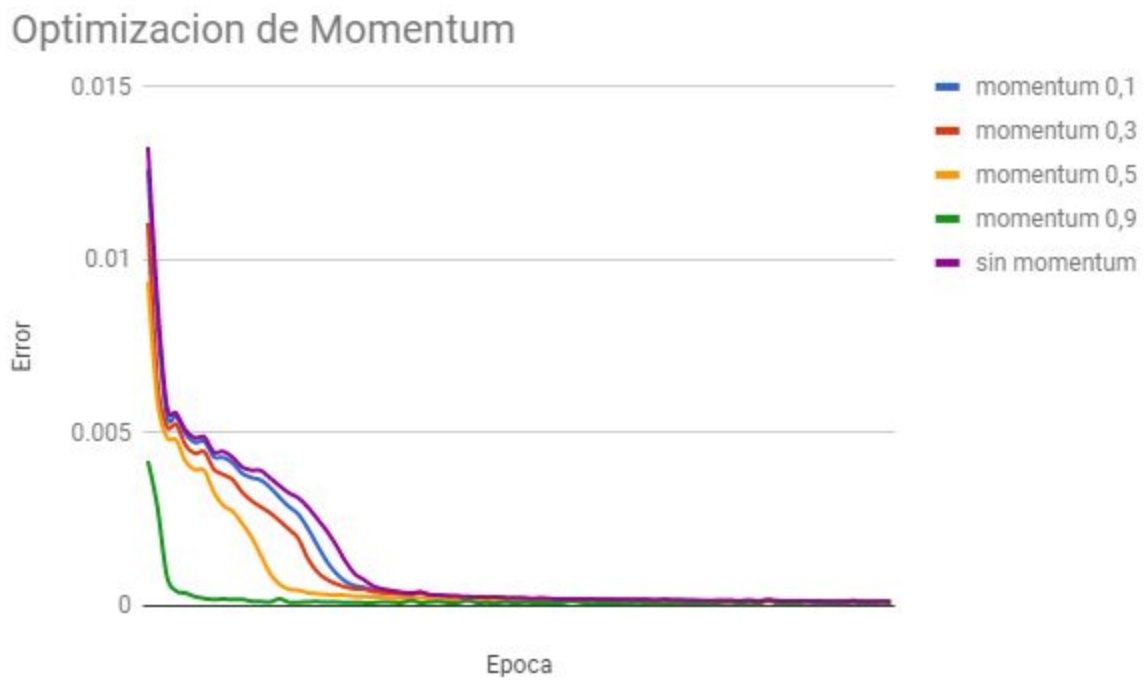
Variación de Eta





Podemos ver como el eta positivo cuan mayor comienza a causar conflictos en la convergencia de la red, posiblemente dado al hecho de que el learning rate se va de rango. Por lo que se recomienda utilizar un eta adaptativo positivo con mayores mucho más chicos dentro del rango de las décimas de unidad. El parámetro del eta adaptativo negativo no se ve que afectará tanto debido a que en incremental el error es errático por lo que sucesivas reducciones de error son menos comunes.

Momentum



El momentum acelera considerablemente la convergencia, esto también se vio en nuestro algoritmo genético cuando todas las iteraciones convergen al mayor valor posible, que habíamos indicado como 0.95.

Conjunto de entrenamiento y capacidad de la red para generalizar

Todos los gráficos presentados en este informe presentan el error cuadrático medio de entrenamiento de la red únicamente. Esto se debe a que se observó constantemente que el error de testeo (error cuadrático medio evaluado únicamente sobre aquellos patrones que no se le presentaban a la red) tendía a los mismos valores que el de entrenamiento, siendo mínima la diferencia con este.

Se busco analizar cuál era la razón por la que esto ocurría y se arribó a que, debido a que el conjunto de puntos de entrenamiento y testeo se seleccionan aleatoriamente, la red entrenada era capaz de predecir con una precisión muy elevada la altura de aquellos puntos distribuidos de testeo, gracias a haber aprendido sus alrededores.

Sin embargo, se realizó una prueba en la que a la superficie se le quito uno de sus picos y se lo separó como conjunto de testeo, fue en este caso donde se vio una discrepancia sustanciosa entre el error de entrenamiento y el de testeo, donde el conjunto de entrenamiento se aprendió muy bien, pero el error de testeo se estancó en cierto punto, debido a la falta de corrección e información de la red en el área en el que este se encontraba.

Ruido

Intentamos agregar ruido al momento de tener un error que no se alteraba después de varias iteraciones, sin embargo como la mayoría de los mínimos locales parecían ser profundos, esto solo lograba que volvieran al mismo error. Si se aumentaba el ruido, el error se modifica a valores mucho mayores y consistía básicamente en volver a entrenar la red, por lo que lo terminamos descartando.

Conclusión

Se concluyo que la red posee un gran poder de generalización siempre y cuando los conjuntos de entrenamiento y testeo se encuentren elegidos de una manera distribuida sobre la superficie, para que la red pueda tener conocimientos sobre los entornos de todos los puntos y poder así estimar correctamente sus valores, sin tener grandes zonas "vacías"en las cuales no tiene soporte para hacer una estimación validera.

La diferencia principal entre batch e incremental, es que el método de batch garantiza un constante decrecimiento al mínimo local alcanzado, mientras que incremental es más errático. Esto resulta en que batch tardará generalmente mayor cantidad de épocas, pero debido al hecho de poder realizar todos los cálculos en una matriz con un simple comando de octave lo compensa un poco. Por eso, si se desea hacer optimizaciones en cuanto a el estado actual del error, batch es una solución más óptima para analizarlo, pero encontramos que incremental nos daba mejor resultado en menor cantidad de tiempo.

En cuanto al momentum, para la aceleración de convergencia es el factor que más afecta el resultado y el de mejor rendimiento. Hemos notado que se recomienda el uso de un valor entre 0.9 y 0.95.

Finalmente podemos decir que, eligiendo muestras dispersas por el terreno y una estructura correcta como definida anteriormente, la red aprenderá en su mayoría de los casos y con el uso de momentum este proceso se acelera considerablemente. Finalmente el eta adaptativo ayuda también a la velocidad aunque en menor magnitud.